**RECURSION**

A function may call itself either directly or indirectly.

#include<stdio.h>

/\* printd: print n in decimal \*/

void printd( int n )

{

if( n < 0 )

{

putchar(‘-’);

n = -n;

}

if( n / 10 )

printd( n / 10 );

putchar( n % 10 + ‘0’ );

}

When a function calls itself recursively, each invocation gets a fresh set of all the automatic variables, independent of the previous set. Thus in **printd**( 123 )the first **printd** receives the argument n =123. It passes 12 to the second **printd**, which in turn passes 1 to a third. The third-level **printd** prints 1, then returns to the second-level. That **printd** prints 2, then returns to first level. That one prints 3 and terminates.

Recursion may not be guaranteed to be faster; but recursive code is more compact, and often much easier to write and understand than the non-recursive equivalent. Recursion is especially convenient for recursively defined data structured like trees.